UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/708,153	02/11/2004	Hsiu-Chuan Lien	MSCP0024USA	2152

27765 7590 12/24/2008
NORTH AMERICA INTELLECTUAL PROPERTY CORPORATION
P.O. BOX 506
MERRIFIELD, VA 22116

EXAMINER	
WEI, ZHENG	
ART UNIT	PAPER NUMBER
2192	

NOTIFICATION DATE	DELIVERY MODE
12/24/2008	ELECTRONIC

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Notice of the Office communication was sent electronically on above-indicated "Notification Date" to the following e-mail address(es):

winstonhsu.uspto@gmail.com
Patent.admin.uspto.Rcv@naipo.com
mis.ap.uspto@naipo.com.tw

Office Action Summary	Application No.	Applicant(s)	
	10/708,153	LIEN ET AL.	
	Examiner	Art Unit	
	ZHENG WEI	2192	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE *3* MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *22 September 2008*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-4,6-8 and 17-21* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☐ Claim(s) *1-4,6-8 and 17* is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☐ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

Detailed Action

Remarks

1. A request for continued examination under 37 CFR 1.114, including the fee set forth in 37 CFR 1.17(e), was filed in this application after final rejection. Since this application is eligible for continued examination under 37 CFR 1.114, and the fee set forth in 37 CFR 1.17(e) has been timely paid, the finality of the previous Office action has been withdrawn pursuant to 37 CFR 1.114. Applicant's submission filed on 09/22/2008 has been entered.
2. This office action is in response to the amendment filed on 09/22/2008.
3. Claims 20 and 21 have been amended.
4. Claims 1-4, 6-8 and 17-21 remain pending and have been examined.

Response to Arguments

5. Applicant's arguments filed on 09/22/2009 have been fully considered but they are not persuasive. For example,
   - At page 7, lines 13-18, the Applicants point out that "...by utilizing Schieve's application even a suggested by the Examiner, a user can only know if existing peripheral 'pass the test' (Fig.4). Peripherals that are not detected are not tested. On the other hand, present claim 1 includes limitations of processing both the general processing path and the error processing path for

each event, regardless if the peripheral device being tested actually exists, something Schieve cannot do."

However, the Examiner respectfully disagrees.

It should be noted that, the claim language does not recite any limitations about processing both the general processing path and the error processing path for each event [emphasis added] as the Applicants argued. It also should be noted that the claim language does not exclude using Schieve's method/steps for detecting peripherals. Moreover, by incorporating Sanchez's method to inject faults and errors (simulate) for testing the reliable and proper handling of various faults and exception under various conditions, it is clear that Schieve's improvement can test the"error state" of the peripherals (error processing path) by simulating the error condition (resetting a parameter to simulate) and "test pass" (general processing path) as recited in the claim 1. Therefore, Schieve and Sanchez together, do disclose the limitation as the Applicants argued.

- At pages 7-8, the Applicants submit that Schieve's application and Sanchez's application should not be combined as suggested. Because Sanchez's application runs on top of operating system in high-level environment as Sanchez disclosed the example for testing Java programming. However, the Examiner's position is that Sanchez discloses a method of automatically injecting faults and errors into a JAVA application or program to test the reliable and proper handing of various faults and exception under various

conditions. The basic idea of Sanchez is injecting faults and errors to simulate the error condition/state for the testing. Therefore, said idea/method can also be incorporated with Schieve's method in step 470 of Fig.4 during executing the test to debug the program for handling different error states. Therefore, the combination of Schieve and Sanchez is reasonable.

Claim Rejections - 35 USC § 103

6. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

(a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

7. Claims 1-2, 7 and 17-21 are rejected under 35 U.S.C. 103(a) as being unpatentable over Schieve (Schieve et al. US 5,463,766) in view of Sanchez (Sanchez et al., US 6,477,666 – art of record)

Claim 1:
Schieve discloses a method for testing and debugging computer programs, the method comprising:

- Setting a plurality of breakpoints corresponding to a plurality of events in an implementation under test, each event being a test executed to a peripheral device (see for example Fig.4, step 410, "Detect Peripherals"" and related text) and taking a general processing path when the peripheral device is working well (see for example, step 480, "Did test pass? Yes" and related

text) or an error processing path when the peripheral device is out of order (see for example, step 490 "Display Error/Status and Information" and related text);

- Executing the implementation under test for outputting a diagnosis code (fig.4, step 480, output YES/NO) of a breakpoint (see for example, step 470, "Execute Test" and related text);
- Resetting a parameter of the event corresponding to the diagnosis code (see for example, Fig.4, step 480, "Did Test Pass?" and "Yes/No" paths and related text; The "Yes/No" parameter has to be reset each time after step 470);
- Executing the event corresponding to the diagnosis code according to the reset parameter for making the event undergo the error processing path (see for example, step 490 "Display Error/Status and Information" and related text);

But Schieve does not explicitly disclose resetting a parameter to simulate the peripheral device being in the error state throughout execution of the event corresponding to the diagnosis code, However, Sanchez in the same analogous art of testing the computer program about reliable and proper handling of various faults under various conditions, discloses a method to simulate the error state throughout execution (injecting faults and error) (see for example, Fig.9, step 70, "Configure Program/Application for automatic fault injection by setting one or more breakpoints within the program/application wherein the breakpoints are where faults may be injected"; step 72, "automatic fault injector is initiated" and related text). Therefore, it would have been obvious to one having ordinary skill in the art at the time the invention was made to use Sanchez's fault injection method to simulate the error state of peripheral device in Schieve. One would have been motivated to do so to test the reliable and proper handling of various faults and exceptions under various conditions as suggested by Sanchez (see for example, ABSTRACT, lines 2-4, "to test the reliable and proper handling of various faults and exceptions under various conditions")

Claim 2:

Schieve and Sanchez disclose the method for program debugging as in claim 1, Sanchez further discloses wherein the method further comprising:

- after executing the event corresponding to the diagnosis code according to the reset parameter for making the event undergo the error processing path for making the implementation under test make all events undergo the error processing path (see for example, see for example, Fig.9, step 70, "Configure Program/Application for automatic fault injection by setting one or more breakpoints within the program/application wherein the breakpoints are where faults may be injected"; step 72, "automatic fault injector is initiated" and related text),
- repeating the steps of executing the implementation under test for outputting the diagnosis code of the breakpoint (see for example, Fig.9, steps 74-80 and col.5, lines 46-50, "When the Event Hook code is executed, then either the Java byte code is allowed to continue or the fault is dynamically inserted. If the latter event is to occur, then the automatic fault injection algorithm is executed by Automatic Fault Injector 30"),
- resetting the parameter of the event corresponding to the diagnosis code (see for example, Fig.6, about resetting attributes parameter, "Fail", "Retry", "Abort"; Fig.7, item 44, "Inject the fault based upon the value of a variable" and related text) and

- executing the event corresponding to the diagnosis code according to the reset parameter for making the event undergo the error processing path (see for example, Fig.9 and col.5, lines 46-50, "When the Event Hook code is executed, then either the Java byte code is allowed to continue or the fault is dynamically inserted. If the latter event is to occur, then the automatic fault injection algorithm is executed by Automatic Fault Injector 30").

Claim 7:

Schieve and Sanchez disclose the method for program debugging as in claim 1 above, which has an error handler to display error message (see for example, Fig.4, step 490, "Display Error /Status and Information"). Schieve also discloses a step of system reset (see for example, Fig.3, step 380, "Reset Button Pressed?"), but does not explicitly disclose the error handler is a system reset. However, it would have been obvious to one having ordinary skill in the art at the time the invention was make to use the method of system reset to handle found error. One would have been motivated to do so to reset the system to prevent whole system crash when some severe bugs occur (see for example, Fig.3 step 380, "Reset Button Pressed?" option "Yes").

Claim 17:

Schieve further discloses the method of claim 1 further comprising:

- executing the implementation under test until the diagnosis code of the breakpoint matches a predetermined diagnosis code before resetting the

parameter of the event corresponding to the diagnosis code (see for example, Fig.9, step 70, step 76 “Has breakpoint occurred within called object code?” and related text) , and executing the event corresponding to the diagnosis code according to the reset parameter for making the event undergo the error processing path (see for example, Fig.9, step 78, “Should a fault be inserted?”; step 80 “pick one of the exceptions for this method and throw it” and related text).

Claims 18-19:

Schieve and Sanchez disclose the same method for program debugging as addressed in Claims 1 and 2 above. All the limitations have been disclosed by Schieve and Sanchez. Therefore, claims 18-19 also would have been obvious in view of reference teachings above. For example, the method comprising:

- setting a plurality of breakpoints corresponding to a plurality of events in an implementation under test, each event being a test executed to a peripheral device (see for example, Schieve , Fig.4, steps 410-490 and related text) and taking a general processing path when the peripheral device is working well (see for example, Fig.9, step 78, "Should a fault be inserted?" path "No" and related text) or an error processing path when the peripheral device is in an error state (see for example, Sanchez , Fig.9, step 70, step 78, "Should a fault be inserted?" path "Yes" and related text);

- setting a parameter to simulate that the peripheral device is working well throughout execution of the implementation under test (see for example, Sanchez, Fig.9, step 78, “Should a fault be inserted?” and related text);
- executing the implementation under test according to the parameter for outputting a diagnosis code corresponding to each breakpoint (see for example, (see for example, Fig.9, step 78, step 80 and related text) ;
- for each breakpoint, determining whether the diagnosis code matches a user defined diagnosis code (see for example, Sanchez, Fig.9, steps, 76 “has breakpoint occurred within called object code?”, step 78, “Should a fault be inserted?” and related text); and
- resetting the parameter to simulate that the peripheral device is in the error state and executing the event corresponding to the diagnosis code according to the reset parameter for making the event undergo the error processing path when it is determined that the diagnosis code matches the user defined diagnosis code (see for example, Sanchez, Fig.9, step 78, “Should a fault be inserted?”, path “Yes”; step 80, “Pick one of the exceptions for this method and throw it” and related text).
- continuing execution of the implementation under test to a next breakpoint without resetting the parameter when it is determined that the diagnosis code does not match the user defined diagnosis code (see for example, Sanchez, Fig.9, step 78, "Should a fault be inserted?" path "No" and related text).

Claims 20-21:

Schieve and Sanchez disclose the same method for program debugging as addressed in Claims 18 and 19 above. All the limitations have been disclosed by Schieve and Sanchez. Therefore, claims 20 and 21 also would have been obvious in view of reference teachings above.

8. Claims 3-4 and 8 are rejected under 35 U.S.C. 103(a) as being unpatentable over Schieve (Schieve et al. US 5,463,766) in view of Sanchez (Sanchez et al., US 6,477,666) and in further view of Phillips (Phillips et al., US 5,321,828)

Claim 3-4:

Schieve discloses the method for program debugging as in claim 1 above, but does not explicitly disclose the breakpoints are set ahead of program codes of the corresponding events or after program codes of the corresponding events. However, Phillips in the same analogous art of an in-circuit emulator for hardware/software development and debugging microprocessors discloses that a user to set any number of breakpoints all at the same place in the program, or at different places (see for example, col.26-col.27, section "Setting Breakpoints" and related descriptions). Therefore, it would have been obvious to one having ordinary skill in the art at the time the invention was made to set breakpoints anywhere in the code in order to adequately support execution control functionality and provide the rich set of functionality needed for the debugger. One would have been motivated to set breakpoints before or after the program

codes of the corresponding events to narrow down the places where the bugs might occur.

Claim 8:

Schieve and Sanchez disclose the method for program debugging as in claim 1 above which has an error handler to display error message, but do not explicitly disclose the error handler is a system execution interrupt. However, Phillips in the same analogous art of an in-circuit emulator for hardware/software development and debugging microprocessors discloses that execution interrupt (see for example, col.72, lines 60-67, "single interrupt request line"). Therefore, it would have been obvious to one having ordinary skill in the art at the time the inversion was make to use the method of system execution interrupt to allow the control processor to monitor the Clock Detect signals which is suggested by Phillips. One would have been motivated to do so to stop executing or suspend current process to trace the problem.

9. Claim 6 is rejected under 35 U.S.C. 103(a) as being unpatentable over Schieve (Schieve et al. US 5,838,975) in view of Sanchez (Sanchez et al., US 6,477,666) and in further view of Robinson (Jeffrey I. Robinson, US 5,768,591)

Claim 6:

Schieve and Sanchez disclose the method for program debugging as in claim 1 above, but do not explicitly disclose that the error handler is an audible tone. However, Robinson discloses a similar method for program debugging as in

claim 1 above which the error handler is an audible tone. (Fig.4, items 172, 164, col.12, lines 64-67, "A sound generator 164 is provided and controlled by the message parser and error handler 172"). Therefore, it would have been obvious to one having ordinary skill in the art at the time the invention was made to use "sound generator" to replace Schieve's method of error handler. One would have been motivated to do so to generate alarm to alert the user when the bug occurs.

Conclusion

10. The prior art made of record and not relied upon is considered pertinent to applicant's disclosure.

11. Any inquiry concerning this communication or earlier communications from the examiner should be directed to Zheng Wei whose telephone number is (571) 270-1059 and Fax number is (571) 270-2059. The examiner can normally be reached on Monday-Thursday 8:00-15:00.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Tuan Q. Dam can be reached on (571) 272-3695. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Any inquiry of a general nature of relating to the status of this application or proceeding should be directed to the TC 2100 Group receptionist whose telephone number is 571- 272-1000.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

/Z. W./
Examiner, Art Unit 2192

/Tuan Q. Dam/
Supervisory Patent Examiner, Art Unit 2192